Performante Erzeugung von Berichten in (vor)formatierten Excel-Dateien

Stephan Frenzel,Sarah Baumert
Robert Koob
KYBEIDOS Gmbh
Moltkestraße 20
69120 Heidelberg

Ralf Neumüller
beyondbytes
Brunnenstraße 60
64372 Ober-Ramstadt

Zusammenfassung

Berichte werden von Fachabteilungen meist in Form einer formatierten Excel-Datei gewünscht, welche die Richtlinien des Corporate Designs im Unternehmen erfüllt und gegebenenfalls Excel-Formeln, -Diagramme und VBA-Makros enthält. Microsoft Technologien eröffnen neue Möglichkeiten, direkt aus einer SAS/Base-Prozedur formatierte Excel-Dateien zu befüllen. Mittels Microsofts ActiveX Data Objects, aufgerufen aus einer selbst implementierten SAS-Prozedur, werden SQL-Statements ausgeführt, die SAS-Daten in Excel-Arbeitsmappen übertragen. Der Zugriff erfolgt hierbei ausschließlich auf der Basis von Desktop-/Client-Funktionen. Durch diese Methode können 100% SAS-Funktionalität mit 100% Excel-Funktionalität kombiniert werden. Es gibt keine Einschränkungen – jeder denkbare Excel-Report auch mit mehreren Arbeitsblättern inklusive Diagrammen und Makros wird möglich. Die Implementierung dieser Funktionalität in eine SAS/Base-Prozedur besticht durch folgende Vorzüge: hohe Performance, weitreichende Funktionalität, einfache Bedienung, schnelle Implementierung und Lauffähigkeit auch auf kleinen Client-Rechnersystemen.

Schlüsselwörter: Excel-Reporting, SAS, SQL, PROC D2O, Data2Office, Excel, Export

1 Einleitung

Trotz hervorragender Möglichkeiten zur Datenpräsentation mit SAS werden Berichte (Reports) von Fachabteilungen zumeist in Form eines Excel-Sheets gewünscht. Zur Übertragung der Daten aus einem SAS Data Set nach MS-Excel stehen die verschiedensten Methoden bereit:

- SAS/Base-Methoden zum unformatierten Export nach MS-Excel
  - Excel Libname Engine
  - PROC EXPORT
  - PROC PRINTTO
  - Data Step mit FILE und PUT
  - ODS CSV
- SAS/Base-Methoden zum formatierten Export nach MS-Excel
  - ODS HTML
  - ODS Tagsets.Msoffice2k (über HTML)

    - ODS Tagsets.ExcelXp (über XML)
    - ODS Tagsets.Tableeditor
- Methoden, die Microsoft-Technologien verwenden
    - DDE
    - ODBC
    - IOM, ADO, VBA
- SAS-BI-Technologien
    - SAS Add-In for Microsoft Office
- SAS/AF
- Java Technologien über JDBC
    - Actuate eSpreadsheet (Projekt BIRT)
    - Apache Jakarta POI Projekt

Literatur zu diesen Technologien liegt in großem Umfang im Internet vor. Einfaches Googlen (export sas data to excel), ein Blick in das SAS-Wiki oder das SAS-Anwenderhandbuch der Uni Heidelberg helfen hier weiter. Eine kleine Literatursammlung befindet sich am Ende dieses Artikels. All diese Methoden bieten jedoch nicht – oder nur sehr beschränkt – die Möglichkeit mit einfachen SAS/Base-Programmen vorformatierte Excel-Templates zu befüllen. Um diese Lücke zu schließen, wurde die native SAS-Prozedur Data2Office (PROC D2O) entwickelt.

2 Data2Office

Data2Office bietet eine neue Technik, um aus SAS heraus Excel-Arbeitsmappen zu generieren. Das Konzept von Data2Office für SAS besteht darin, aus Excel-Vorlagen per SQL auf SAS-Dateien zuzugreifen – und das ohne den Umweg über SAS-Server (z.B. Share-Server). Der Zugriff erfolgt ausschließlich auf der Basis von Desktop-/Client-Funktionen. Data2Office läuft unter allen Windows NT basierten Betriebssystemen (Windows NT, 2000, XP, Vista, 7) und ist kompatibel mit SAS 8.2 und höher sowie MS-Excel 97 und höher.
Der Data2Office-Kern ist schon seit Jahren in einer Reihe von Projekten erprobt. Dessen Bereitstellung in Form einer SAS Prozedur übertrifft bestehende Lösungen durch folgende Vorzüge:

- Hohe Performance (Massenbericht-Erzeugung durch Batch-Verarbeitung)
- Weitreichende Funktionalität (SAS-Datenimport in Grafiken, Formeln und VBA-Makros enthaltende, formatierte Excel- Arbeitsmappen)
- Einfache Bedienung (native SAS-Prozedur)
- Schnelle Implementierung (SAS-Base und SQL-Kenntnisse ausreichend)
- Ausgabe von Reports als PDF- und HTML-Dokumente unter Microsoft Office 2010

3 Funktionsweise

Die SAS-Prozedur von Data2Office besteht aus dem Kern-Modul D2O.exe, das in einer SAS-Prozedur gekapselt wurde. D2O.exe wurde mit Delphi unter Zuhilfenahme von Assembler-Routinen entwickelt. Die Entwicklung der SAS-Prozedur wurde in C auf Basis von SAS/Toolkit vorgenommen. Wird die SAS-Prozedur aufgerufen, transformiert diese zunächst die im TABLE-Statement angegebenen SAS-Datasets in eine einzige CSV-Tabelle. Das Kern-Modul D2O.exe liest diese ein und befüllt damit eine temporäre Microsoft SQL-Datenbank. Auf diese Datenbank erfolgt dann der Zugriff aus dem Excel-Sheet heraus. Hierzu durchsucht Data2Office alle Blätter einer Excel-Arbeitsmappe bis es auf den Tag D2OSQL trifft. Der dort angegebene SQL-Befehl wird auf der temporären Datenbank abgesetzt und die Ergebnisdaten in das Excel-Sheet übertragen (s. **Abbildung 1**). Im Anschluss an die Befüllung wird ein ggf. vorliegendes VBA-Makro „d2oopen“ ausgeführt.

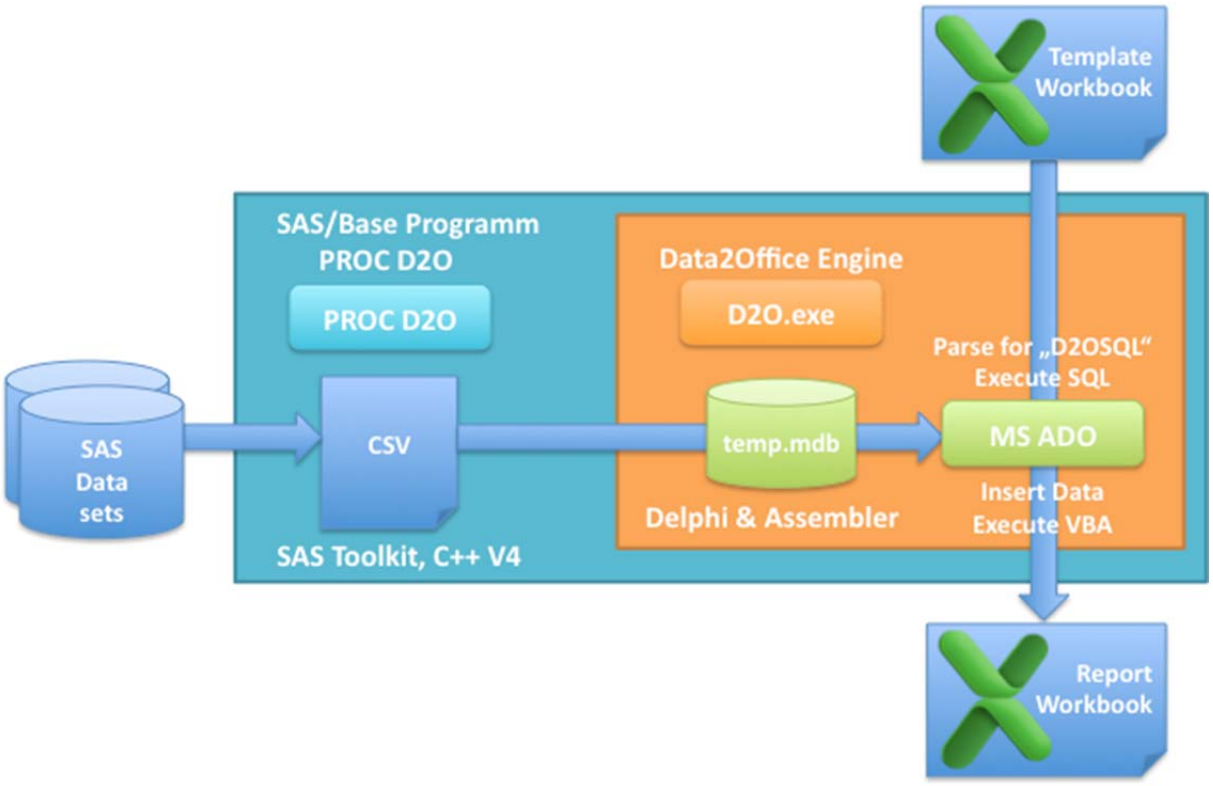

Abbildung 1: Funktionsweise von Data2Office

4 Syntax

Die Prozedur D2O besteht aus den zwei Statements PROC D2O und TABLES. Mit dem PROC D2O-Statement wird das zu verwendende Excel-Template (TEMPLATE =) und die zu erzeugende Excel-Ausgabe-Datei (OUTPUT =) festgelegt. Das TABLES-Statement weist den zu exportierenden SAS-Dateien (sas-dataset*n*) den im Excel-Template verwendeten Namen (d2osql-name*n*) zu.

```
PROC D2O TEMPLATE = “pfad\exceltemplate.xls”
         OUTPUT   = “pfad\exceldatei.xls”
```

```
          <SHOW|NOSHOW>
          ;
   TABLES sas-dataset1 = d2osql-name1
          < … sas-datasetn = d2osql-namen >
          ;
run;
```

Die Optionen SHOW und NOSHOW legen fest, ob die erzeugte Excel-Datei nach dem Ausführen der Prozedur Data2Office automatisch geöffnet wird (SHOW) oder nicht (NOSHOW). NOSHOW ist besonders für den Batch-Betrieb geeignet.

5 Anwendung

Die SAS-Prozedur D2O erzeugt mit SAS-Daten gefüllte Excel-Arbeitsmappen. Die Formatierung der Arbeitsblätter – inklusive Excel-Zellformaten, -Berechnungen, -Makros und -Grafiken – wird hierzu in einer Template-Arbeitsmappe vorgegeben. Die SAS-Prozedur D2O untersucht ein vorgegebenes Excel-Template auf das Schlüsselwort D2OSQL und fügt an dieser Stelle die Daten des angegebenen SAS-Datasets ein.

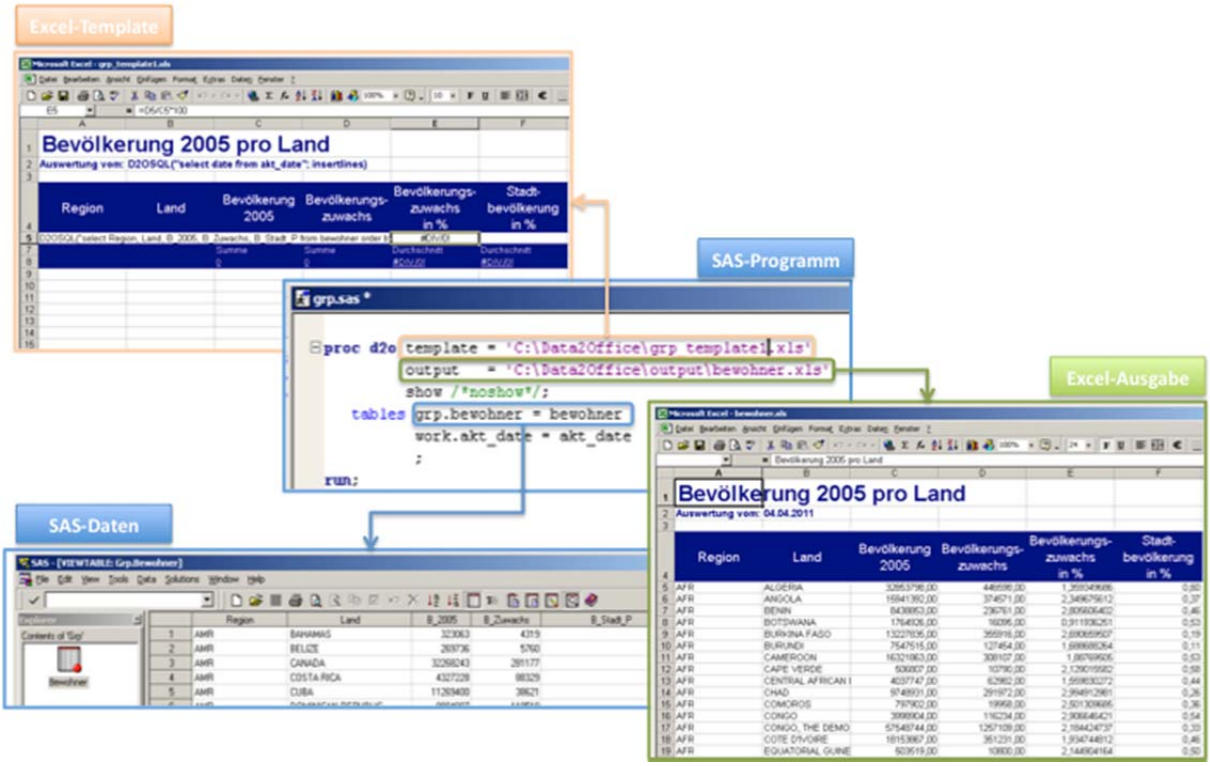

Abbildung 2: Beispiel für die Nutzung von PROC D2O

Folgender Aufruf der SAS-Prozedur D2O stellt die SAS-Datei GRP.BEWOHNER unter dem Namen BEWOHNER für den SQL-Zugriff aus Excel bereit und ruft die D2O-Engine zur Verarbeitung der Excel-Vorlage GRP_TEMPLATE1.XLS auf:

```
proc d2o template = 'C:\Data2Office\grp_template1.xls'
         output   = 'C:\Data2Office\output\bewohner.xls'
```

```
        show;
   tables grp.bewohner = bewohner
          ;
run;
```

Mit der Funktion D2OSQL in der Excel-Vorlage werden die SAS-Daten dann selektiv nach Excel übernommen:

```
D2OSQL("select Region, Land, B_2005, B_Zuwachs, B_Stadt_P
from bewohner order by Region, Land";insertlines)
```

Als Ergebnis wird das Excel-Sheet BEWOHNER.XLS erzeugt, das die in der Funktion D2OSQL angegebenen Daten des SAS Data Sets GRP.BEWOHNER enthält.
Die Funktion D2OSQL kann an mehreren Stellen in einem Tabellenblatt, auf verschiedenen Tabellenblättern, in Kommentaren und in der Definition von Hyperlinks in einem Excel-Dokument eingefügt werden. Die SQL-Befehle können sich dabei auf unterschiedliche SAS-Dateien oder beliebige Verknüpfungen (Joins) von SAS-Dateien beziehen. So können mit einem Aufruf von PROC D2O mehrere SAS-Dateien in eine Excel-Arbeitsmappe übernommen werden. In den Excel-Vorlagen sind alle SQL-Funktionalitäten des Standards ANSI-SQL92 nutzbar.
Bei der Datenübernahme bleiben in Excel berechnete Spalten innerhalb des Bereichs, in den Daten übernommen werden sollen, erhalten; sie werden beim Einfügen der Spalten übersprungen. Dadurch ist es möglich, in eingefügten Tabellen auch in MS-Excel berechnete Spalten zu verwenden.
Zur Steuerung des Einfügens von Zeilen und Spalten stehen verschieden Optionen der Excel-Funktion D2OSQL zur Verfügung:
D2OSQL (<SQL-Befehl>, insertlines) Es werden komplette horizontale Zeilen eingefügt.
D2OSQL (<SQL-Befehl>, insertcells) Es wird nur der notwendige Bereich (Zellen) eingefügt.
D2OSQL (<SQL-Befehl>, noinsert) Das Ergebnis überschreibt die vorhandenen Zellen.
Es werden keine neuen Zeilen oder Zellen eingefügt.
Nach dem Datenimport kann für die Nachbearbeitung der Excel-Arbeitsmappe ein VBA-Makro mit dem vordefinierten Namen D2OOPEN ausgeführt werden. Dieses Makro wird – wenn vorhanden – vor dem Speichern der fertigen Excel-Arbeitsmappe ausgeführt. Damit hat man letztendlich die Möglichkeit, den wirklich vollen Umfang der Excel-Funktionalitäten nach dem Befüllen mit Daten zu nutzen.
Unter Verwendung von MS-Office 2007 und 2010 können Template- und Ausgabe-Arbeitsmappe auch im Open XML Format vorliegen (XLSX und XLTX). Ist ein auszuführendes VBA-Macro D2OOPEN im Template vorhanden, ist das Format XLSM zu verwenden.
Unter MS-Office 2010 kann die Ausgabe auch in einer PDF- oder HTML-Datei erfolgen. Hierzu wird anstelle einer Excel-Datei eine PDF- oder HTM-Datei im Output-Parameter angegeben.

```
output    = 'C:\Data2Office\output\bewohner.pdf'
output    = 'C:\Data2Office\output\bewohner.htm'
```

Die Formatierungsvorgaben des Excel-Druckdialoges werden bei der Ausgabe als PDF-Datei berücksichtigt.

6 Data2Office-Funktionalitäten als Vaadin-App für das SAS-BI-Portal

Um die Data2Office-Funktionalitäten auch im BI-Portal von SAS verfügbar zu machen, steht der D2O-Kern auch in Form eines Vaadin-Portlets zur Verfügung.
Vaadin ist ein serverseitiges Framework. Es kombiniert so die Vorteile der Java-basierten GUI-Entwicklung mit einer Java-IDE. Vaadin unterstützt die Entwicklung von Portlets nach den beiden Standards JSR 168 und JSR 268. Portlets, die in das SAS BI-Portal eingebunden werden sollen, müssen dem JSR 168 Standard entsprechen.
Das unten dargestellte Vaadin-Portlet mit der D2O-Anwendung ermöglicht es, aus einer z. B. mit SAS erzeugte CSV-Datei und einem Excel-Template ein formatiertes Excel zu erstellen.

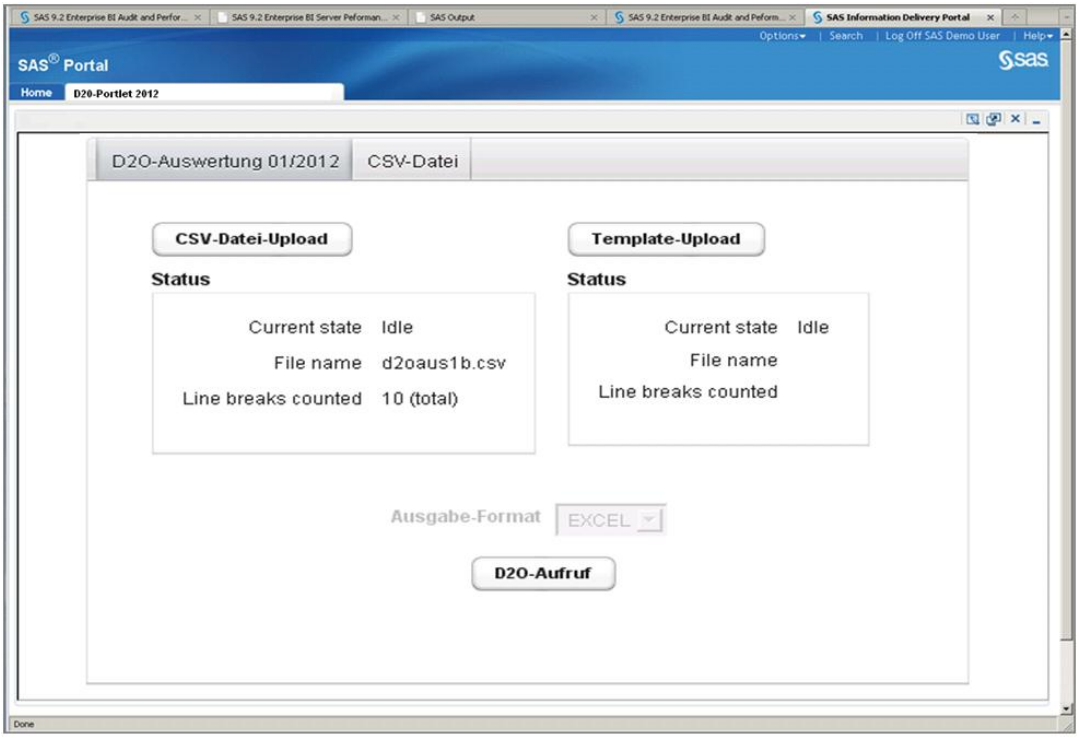

Abbildung 3: Data2Office für das SAS-BI-Portal

Man lädt die gewünschte CSV-Datei und das dazu passende Template vom Client zum Server hoch. Wenn die CSV-Datei hochgeladen wurde, besteht für den Anwender die Möglichkeit die übertragene Datei im Reiter „CSV-Datei“ anzuschauen. Die D2O-Anwendung selbst befindet sich bereits auf dem Server und muss nicht auf dem Client installiert werden. Der Benutzer wählt lediglich das Ausgabe-Format (HTML, PDF oder

Excel) und ruft die D2O-Anwendung auf. Der Output wird erzeugt und angezeigt und kann anschließend abgespeichert werden.

7 Entwicklung eigener SAS-Prozeduren

Die Entwicklung eigener SAS-Prozeduren erfolgt auf Basis des Produkts SAS/Toolkit. Neben der Entwicklung von SAS-Prozeduren unterstützt SAS/Toolkit auch die Entwicklung von Formaten, Funktionen und CALL-Routinen.

Das Schreiben einer eigenen SAS-Prozedur besteht im Kern aus den folgenden zwei Schritten:

1. Erstellung einer schematisierten Definition der Syntax der Prozedur – der s.g. "Grammatik" (engl. Grammar).
2. Schreiben des Programms, das die Verarbeitungslogik der Prozedur implementiert.

Der fraglos angenehmere Teil der Entwicklung einer Prozedur ist das Schreibens der Grammar; hier kann man seiner Phantasie freien Lauf lassen und definieren, welche Anweisungen und Optionen die Prozedur unterstützen soll, d.h. wie Benutzer die Prozedur aufrufen können.

Im Folgenden ist die Grammar-Datei der Prozedur D2O wiedergegeben:

```
#-------------------------------------------------------------#
#    NAME:         EXTRACT                                    #
#    TYPE:         GRAMMAR                                    #
#     PURPOSE:     FRAMMAR  FILE  FOR  PROC  D2O              #
#    END                                                      #
#-------------------------------------------------------------#
%INCLUDE                                                 STUBGRM.
PROGRAM                                  =          ANYSTMT        ENDJB,
ANYSSTMT                                          =                D2OSTMT
          |                                                       TABLESSTMT
          ,
#------------STATEMENT                                  DEFINITIONS----------------#
D2OSTMT   =                                                       @PROCINIT
                              @STMTINIT(10)
            „D2O“                                                 D2OOPTS*
                              @STMTEND
          ,
D2OOPTS                                                                 =
           („TEMPLATE“    „=„                                  @PARM(1,5)    QS)
          |(„OUTPUT“           „=„                             @PARM(2,5)    QS)
          |(„SHOW“                                                    @OPT(1))
          |(„NOSHOW“                                                  @OPT(2))
          ,
TABLESSTMT  =                                                   @STMTINIT(1)
            „TABLES“                                  (@LIST(1,6)   ANYTHING)*
                              @STMTEND                                    ,
ANYTHING     =  INT  |  NBR  |  NAMEX  |  VLITERAL  |  FMT  |  COMPOUND  |  QS  |  SPECIAL  ,
SPECIAL        =  „!“  |  „@“  |  „$“  |  „%“  |  „^“  |  „&“  |  „*“  |  „(„  |  „)“  |
            „-“  |  „+“  |  „=„  |  „|“  |  „:“  |  „<„  |  „>“  |  „?“  |  „/“
.
```

Man erkennt ohne Weiteres, wie hier die Syntax von PROC D2O definiert wird, die dann in Aufrufen wie dem folgenden zum Einsatz kommt:

```
proc d2o template = 'C:\Data2Office\grp_template1.xls'
         output   = 'C:\Data2Office\output\bewohner.xls'
         show;
```

```
   tables grp.bewohner = bewohner
          ;
run;
```

Ein hartes Stück Arbeit verglichen mit der Definition der Grammatik ist dagegen das Schreiben des Programms, das die Funktionalität der Prozedur implementiert.
Für die Implementierung der Funktionalität eigener SAS-Prozeduren stellt SAS/Toolkit eine Reihe von Unterprogrammen zur Verfügung, die wichtige Schnittstellenfunktionen bereitstellen – u.a. für folgende Aufgaben:

- Lesen und Schreiben von SAS-Dateien
- Ermittlung von Information zu Variablen und Datensätzen in SAS-Dateien
- Schreiben von Output oder Meldungen in den SAS-Log
- Handhabung SAS-spezifischer Aspekte, wie z. B. fehlender Werte

SAS/Toolkit unterstützt die Implementierung eigener Prozeduren in einer der folgenden Sprachen: C, Fortran, PL/I und IBM 370 Assembler.
Die folgende Abbildung zeigt die ersten Zeilen des Verarbeitungsprogramms der Prozedur D2O; PROC D2O ist in C implementiert.

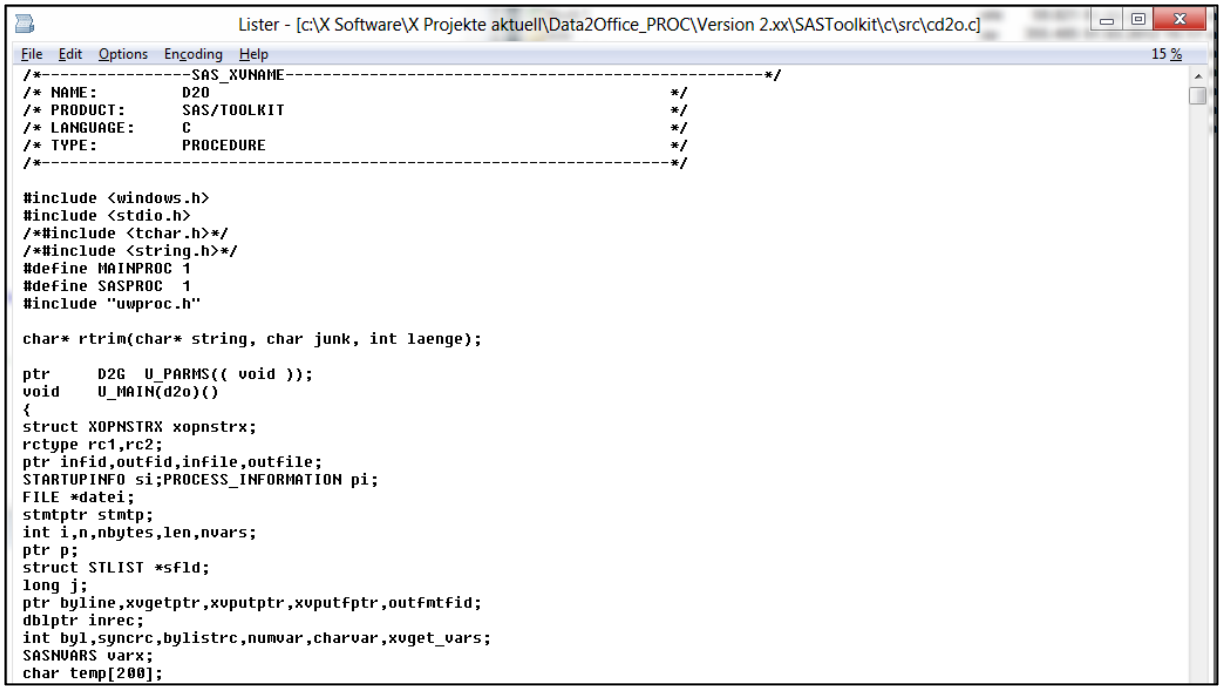

Abbildung 4: Source-Code der SAS-Prozedur D2O

Die folgende Abbildung stellt den Ablauf zur Erzeugung des ausführbaren Programms einer SAS-Prozedur schematisch dar:

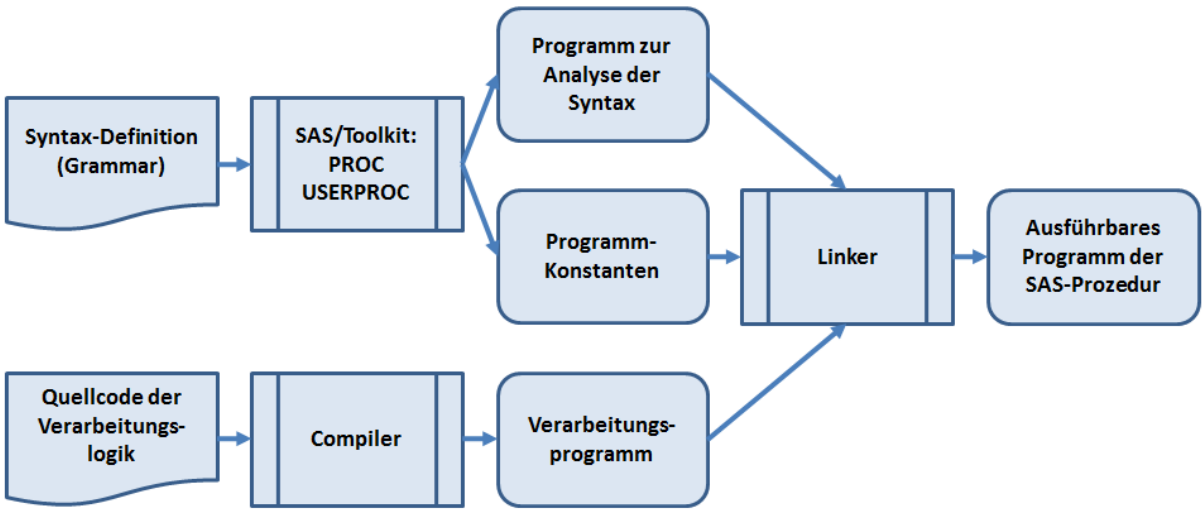

Abbildung 5: Ablauf der Erzeugung einer SAS-Prozedur

Dieser Ablauf bei der Erzeugung des ausführbaren Programms der SAS-Prozedur findet sich dann auch in der sogenannten Make-Datei:

```
########################################################################
#   Copyright   (c)   2008   -   2012   by   KYBEIDOS   GmbH,   Heidelberg,   Germany   #
#  NAME:  prcd2o.make                                                                    #
#  PRODUCT:  SAS/TOOLKIT                                                                 #
#   PURPOSE:   make   file   for   creating   PROC   D2O,   Version   2.0                 #
########################################################################
!include                                                                  toolkit.mak

#                                       local                                    macros
UWOBJ                                   =                                $(OBJ)cd2o.obj
MODNAME                                       =                                    d2o
FUNCFILE                                      =                                  cd2g
GRAMFUNC                                       =                                 d2g
PGMCON                                      =                                   mcbd2o
WHICH                                        =                                    PROC
LANGUAGE                                        =                                  c
PRCINT                                    =                                  $(PRCINTCV)

!include                                                                    where.mak

#                         local                         MAKE                        definitions
ALL                                                                :     $(LOAD)$(MODNAME).dll
$(OBJ)cd2o.obj      : $(SRC)cd2o.c ; \                                    (1)
             $(UWC)           -Fo$(OBJ)cd2o.obj          $(UWCOPTS)           $(SRC)cd2o.c

!include grmfunc.mak                                                      (2)
!include                                                                    pgmcon.mak
!include uwlnkprc.mak                                                     (3)
```

(1) Hier wird definiert, dass das Programm, das die Verarbeitungslogik von PROC D2O implementiert, aus der Quellcode-Datei „cd2o.c“ erstellt wird und wie ggf. der Compiler zur Erzeugung des ausführbaren Programms aufgerufen wird.
(2) Innerhalb der eingebundenen Routinen „grmfunc.mak“ und „pgmcon.mak“ werden die Module zur Analyse der Prozedur-Syntax mit Hilfe der Prozedur USERPROC generiert und kompiliert.
(3) Abschließend wird die Routine “uwlnkprc.mak” aufgerufen, um alle Module zu einem ausführbaren Programm zu verbinden.

Das Ergebnis dieses Prozesses ist die Datei “d2o.dll”, die bei der Installation von Data2Office in das Installationsverzeichnis von SAS gestellt wird.

8 Entwicklungsaufwand und Komplexität

Abschließend einige wenige Worte zur Komplexität und zum Entwicklungsaufwand des Kerns von Data2Office und der SAS-Prozedur D2O: Der Entwicklungsaufwand für beide Komponenten von Data2Office in Relation zu der Anzahl der Lines of Code stellt sich wie folgt dar:

- Data2Office-Kern: Delphi, 803 Zeilen / 8 Mannmonate
- SAS-Prozedur: C-Programm, 286 Zeilen / 4 Mannmonate

Die folgende Abbildung visualisiert diese Relationen:

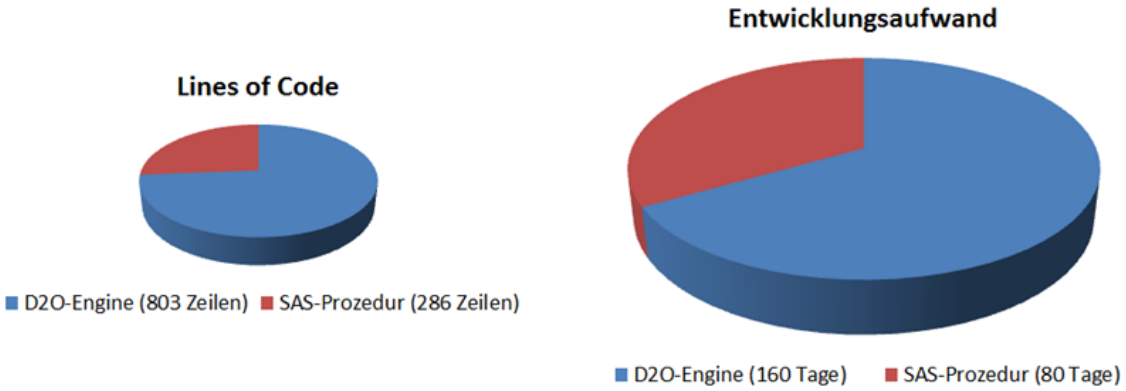

Abbildung 6: Entwicklungsaufwand und Komplexität

Es ist offensichtlich, dass der Entwicklungsaufwand im Vergleich zum Umfang des Source-Codes weitaus höher als bei „durchschnittlichen“ Entwicklungsprojekten ist. Dies ist darauf zurückzuführen, dass sowohl bzgl. des optimierten Ladens von Daten in Excel-Sheets als auch beim Schreiben der SAS-Prozedur erheblicher Forschungsaufwand zu erbringen war. Der entstandene Source-Code ist demgegenüber ausgesprochen kompakt.
Sowohl Data2Office selbst als auch Projekte, die Data2Office nutzen, zeichnen sich also durch Effizienz und Wartbarkeit bei gleichzeitig großem Funktionsumfang aus.

9 Download

Die "Data2Office"-Prozedur "D2O" ist ein Gemeinschaftsprojekt von KYBEIDOS (Konzeption und SAS-Prozedur, http://www.kybeidos.de) und Ralph Neumüller (Data2Office-Kern, http://www.beyondbytes.de).

Eine Test-Lizenz von Data2Office steht auf der Website von KYBEIDOS bereit (http://www.kybeidos.de/loesung/excel-reporting/). Diese enthält folgende Dateien:

- Installer-Datei mit D2O-Prozedur und –Engine
- Beispielprogramm mit D2O-Aufruf
- Mehrere Beispiel-Excel-Templates zum Befüllen mit SAS-Daten
- Handbuch

Literatur

[1] http://www.lexjansen.com/pharmasug/2006/posters/po13.pdf

[2] http://www.nesug.org/proceedings/nesug04/io/io09.pdf

[3] http://www.urz.uni-heidelberg.de/imperia/md/content/urz/programme/statistik/sas-treff/2006-07-07.pdf

[4] http://www.sas.com/offices/europe/austria/html/events/sas_club/downloads/sas_club19_bernadette.pdf

[5] http://www.sas-consultant.com/professional/SEUGI18-Using-DDE-to-Pour-S.pdf

[6] http://www.urz.uni-heidelberg.de/statistik/sas-ah/01.01.03/odbc/odbc.html

[7] http://rfd.uoregon.edu/files/rfd/StatisticalResources/data_trnsf2.txt

[8] http://www2.sas.com/proceedings/forum2007/168-2007.pdf

[9] http://support.sas.com/rnd/papers/sugi29/ExcelXML.pdf

[10] http://www.urz.uni-heidelberg.de/imperia/md/content/urz/programme/statistik/sas-treff/2005-05-13.pdf

[11] http://www.urz.uni-heidelberg.de/statistik/sas-ah/01.01.03/odbc/odbc.html

[12] http://analytics.ncsu.edu/sesug/2008/SIB-105.pdf

[13] http://www2.sas.com/proceedings/sugi30/157-30.pdf

[14] http://support.sas.com/resources/papers/proceedings10/003-2010.pdf

[15] http://www.phuse.eu/download.aspx?type=cms&docID=254

[16] http://support.sas.com/resources/papers/proceedings11/003-2011.pdf